


[Subscribe \(Full Service\)](#) [Register \(Limited Service, Free\)](#) [Login](#)

 Search: ☒ The ACM Digital Library ☐ The Guide



[Feedback](#) [Report a problem](#) [Satisfaction survey](#)

 Terms used **load predicted cache miss compiler**

Found 12 of 145,831

Sort results by


[Save results to a Binder](#)
[Try an Advanced Search](#)
[Try this search in The ACM Guide](#)

Display results


[Search Tips](#)
☐ Open results in a new window

Results 1 - 12 of 12

 Relevance scale ☐ ☐ ☐ ☐ ☐

### 1 [Predictive techniques for aggressive load speculation](#)

Glenn Reinman, Brad Calder

 November 1998 **Proceedings of the 31st annual ACM/IEEE international symposium on Microarchitecture**

Full text available: pdf(1.94 MB)

 Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)


### 2 [Predictor-directed stream buffers](#)

Timothy Sherwood, Suleyman Sair, Brad Calder

 December 2000 **Proceedings of the 33rd annual ACM/IEEE international symposium on Microarchitecture**

Full text available: pdf(187.89 KB)

ps(1.12 MB)

 Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)
[Publisher Site](#)


### 3 [Microarchitectures: Scaling the issue window with look-ahead latency prediction](#)

Yongxiang Liu, Anahita Shayesteh, Gokhan Memik, Glenn Reinman

 June 2004 **Proceedings of the 18th annual international conference on Supercomputing**

Full text available: pdf(302.66 KB)

 Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

In contemporary out-of-order superscalar design, high IPC is mainly achieved by exposing high instruction level parallelism (ILP). Scaling issue window size can certainly provide more ILP; however, future processor scaling demands threaten to limit the size of the issue window. In this study, we propose a dynamic instruction sorting mechanism that provides more ILP without increasing the size of the issue window. In our approach, early in the pipeline, we predict how long an instruction needs to ...



**Keywords:** CLP, LHT, MNM, SILO, instruction sorting



### 4 [Microarchitecture support for improving the performance of load target prediction](#)

Chung-Ho Chen, Akida Wu

 December 1997 **Proceedings of the 30th annual ACM/IEEE international symposium on Microarchitecture**


Full text available:  [pdf\(882.37 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)  
 [Publisher Site](#)

Presents a load target prediction scheme that mitigates the impact of load latency for modern microprocessors. The scheme uses a cache-like buffer to provide the base address, offset and operand size at the instruction fetching stage of a pipeline so that a load target address can be computed earlier at the decode stage. With the dynamic use of a load stride, the scheme has achieved a prediction rate that is 15% higher than a previously proposed approach. By providing a 128-entry direct-mapped l ...

**Keywords:** load target prediction, load-use stall, pipeline, speculative data access, superscalar processor

##### 5 Examination of a memory access classification scheme for pointer-intensive and numeric programs

Luddy Harrison

January 1996 **Proceedings of the 10th international conference on Supercomputing**

Full text available:  [pdf\(991.11 KB\)](#) Additional Information: [full citation](#), [references](#), [citations](#), [index terms](#)

**Keywords:** CPU architecture, data cache, instruction profiling, memory access pattern classification, memory latency tolerance

##### 6 Architecture: Continual flow pipelines

Srikanth T. Srinivasan, Ravi Rajwar, Haitham Akkary, Amit Gandhi, Mike Upton

October 2004 **Proceedings of the 11th international conference on Architectural support for programming languages and operating systems**

Full text available:  [pdf\(274.26 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)



Increased integration in the form of multiple processor cores on a single die, relatively constant die sizes, shrinking power envelopes, and emerging applications create a new challenge for processor architects. How to build a processor that provides high single-thread performance and enables multiple of these to be placed on the same die for high throughput while dynamically adapting for future applications? Conventional approaches for high single-thread performance rely on large and complex co ...

**Keywords:** CFP, instruction window, latency tolerance, non-blocking

##### 7 Speculation techniques for improving load related instruction scheduling

Adi Yoaz, Mattan Erez, Ronny Ronen, Stephan Jourdan

May 1999 **ACM SIGARCH Computer Architecture News , Proceedings of the 26th annual international symposium on Computer architecture**, Volume 27 Issue 2

Full text available:  [pdf\(164.15 KB\)](#) Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#)  
 [Publisher Site](#)

State of the art microprocessors achieve high performance by executing multiple instructions per cycle. In an out-of-order engine, the instruction scheduler is responsible for dispatching instructions to execution units based on dependencies, latencies, and resource availability. Most existing instruction schedulers are doing a less than optimal job of scheduling memory accesses and instructions dependent on them, for the following reasons:• Memory dependencies cannot be resolved prior ...

**8** Value prediction in VLIW machines

Tarun Nakra, Rajiv Gupta, Mary Lou Soffa

May 1999 **ACM SIGARCH Computer Architecture News , Proceedings of the 26th annual international symposium on Computer architecture**, Volume 27 Issue 2Full text available:  pdf(226.09 KB)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#) [Publisher Site](#)

The performance of VLIW architectures is dependent on the capability of the compiler to detect and exploit instruction-level parallelism during instruction scheduling. To exploit the detected parallelism, instructions are reordered to reduce the length of the code schedule and minimize the cycle count for execution. Code reordering is limited by the dependencies among instructions arising from both control flow and data flow. In this paper, we present the design of a VLIW architecture that uses ...

**9** Clustered microarchitectures: Cluster prefetch: tolerating on-chip wire delays in clustered microarchitectures

Rajeev Balasubramanian

June 2004 **Proceedings of the 18th annual international conference on Supercomputing**Full text available:  pdf(153.44 KB)Additional Information: [full citation](#), [abstract](#), [references](#), [index terms](#)

The growing dominance of wire delays at future technology points renders a microprocessor communication-bound. Clustered microarchitectures allow most dependence chains to execute without being affected by long on-chip wire latencies. They also allow faster clock speeds and reduce design complexity, thereby emerging as a popular design choice for future microprocessors. However, a centralized data cache threatens to be the primary bottle-neck in highly clustered systems. The paper attempts to id ...

**Keywords:** clustered microarchitectures, communication-bound processors, data prefetch, distributed caches, effective address and memory dependence prediction, processor

**10** Detecting global stride locality in value streams

Huiyang Zhou, Jill Flanagan, Thomas M. Conte

May 2003 **ACM SIGARCH Computer Architecture News , Proceedings of the 30th annual international symposium on Computer architecture**, Volume 31 Issue 2Full text available:  pdf(292.41 KB)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

Value prediction exploits localities in value streams. Previous research focused on exploiting two types of value localities, computational and context-based, in the local value history, which is the value sequence produced by the same instruction that is being predicted. Besides the local value history, value locality also exists in the global value history, which is the value sequence produced by all dynamic instructions according to their execution order. In this paper, a new type value local ...

**11** Correlated load-address predictors

Michael Bekerman, Stephan Jourdan, Ronny Ronen, Gilad Kirshenboim, Lihu Rappoport, Adi Yoaz, Uri Weiser

May 1999 **ACM SIGARCH Computer Architecture News , Proceedings of the 26th annual international symposium on Computer architecture**, Volume 27 Issue 2Full text available:  pdf(149.18 KB)Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#), [index terms](#) [Publisher Site](#)

As microprocessors become faster, the relative performance cost of memory accesses increases. Bigger and faster caches significantly reduce the absolute load-to-use time delay. However, increase in processor operational frequencies impairs the relative load-to-use


latency, measured in processor cycles (e.g. from two cycles on the Pentium® processor to three cycles or more in current designs). Load-address prediction techniques were introduced to partially cut the load-to-use latency. Thi ...

**Keywords:** context-based predictor, global correlation, load-address prediction, predictor implementation, recursive data structures

## 12 Enhancing software reliability with speculative threads

Jeffrey Oplinger, Monica S. Lam

October 2002 **Proceedings of the 10th international conference on Architectural support for programming languages and operating systems**, Volume 37 , 36 , 30 Issue 10 , 5 , 5

Full text available:  pdf(1.47 MB)

Additional Information: [full citation](#), [abstract](#), [references](#), [citations](#)

This paper advocates the use of a monitor-and-recover programming paradigm to enhance the reliability of software, and proposes an architectural design that allows software and hardware to cooperate in making this paradigm more efficient and easier to program. We propose that programmers write monitoring functions assuming simple sequential execution semantics. Our architecture speeds up the computation by executing the monitoring functions speculatively in parallel with the main computation. For ...

Results 1 - 12 of 12

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2004 ACM, Inc.  
[Terms of Usage](#) [Privacy Policy](#) [Code of Ethics](#) [Contact Us](#)

Useful downloads:  [Adobe Acrobat](#)  [QuickTime](#)  [Windows Media Player](#)  [Real Player](#)

## WEST Search History





DATE: Saturday, November 20, 2004

<u>Hide?</u>	<u>Set Name</u>	<u>Query</u>	<u>Hit Count</u>
	<i>DB=PGPB,USPT; PLUR=NO; OP=ADJ</i>		
<input type="checkbox"/>	L16	5903749.uref.	7
<input type="checkbox"/>	L15	5903749.pn.	1
<input type="checkbox"/>	L14	L3 same compil\$3	1
	<i>DB=EPAB,JPAB,DWPI,TDBD; PLUR=NO; OP=ADJ</i>		
<input type="checkbox"/>	L13	L11 and (repair\$3 or restor\$3)	0
<input type="checkbox"/>	L12	L11 and cache miss	1
<input type="checkbox"/>	L11	predicted load	105
<input type="checkbox"/>	L10	mispredict\$4 with load	4
	<i>DB=PGPB,USPT; PLUR=NO; OP=ADJ</i>		
<input type="checkbox"/>	L9	L5 same (repair\$3 or restor\$3)	15
<input type="checkbox"/>	L8	L6 not L2	4
<input type="checkbox"/>	L7	L6 not L4	4
<input type="checkbox"/>	L6	L5 same (cache miss)	4
<input type="checkbox"/>	L5	mispredict\$4 with load	354
<input type="checkbox"/>	L4	L3 same (cache miss)	3
<input type="checkbox"/>	L3	predicted load	186
<input type="checkbox"/>	L2	cache same (load\$3 with mispredict\$3) same (repair\$3 or restor\$3)	4
<input type="checkbox"/>	L1	cache same load\$3 same mispredict\$3 same (repair\$3 or restor\$3)	153

END OF SEARCH HISTORY